# ADC with STM32F0

An ADC (Analog-To-Digital) converter is an electronic circuit that takes in an analog voltage as input and converts it into digital data. This value represents the voltage level in binary code. The ADC samples the analog input whenever you trigger it to start conversion. It performs a quantization process to decide on the voltage level and its binary code that gets pushed into the output register.

In this tutorial, we learn about the STM32 ADC Polling method.

## STM32 ADC Polling method:

It's the easiest way in code to perform an analog-to-digital conversion using the ADC on an analog input channel. However, it's not an efficient way in all cases as it's considered to be a blocking way of using the ADC. In this way, we start the A/D conversion and wait for the ADC until it completes the conversion so the CPU can resume processing the main code.

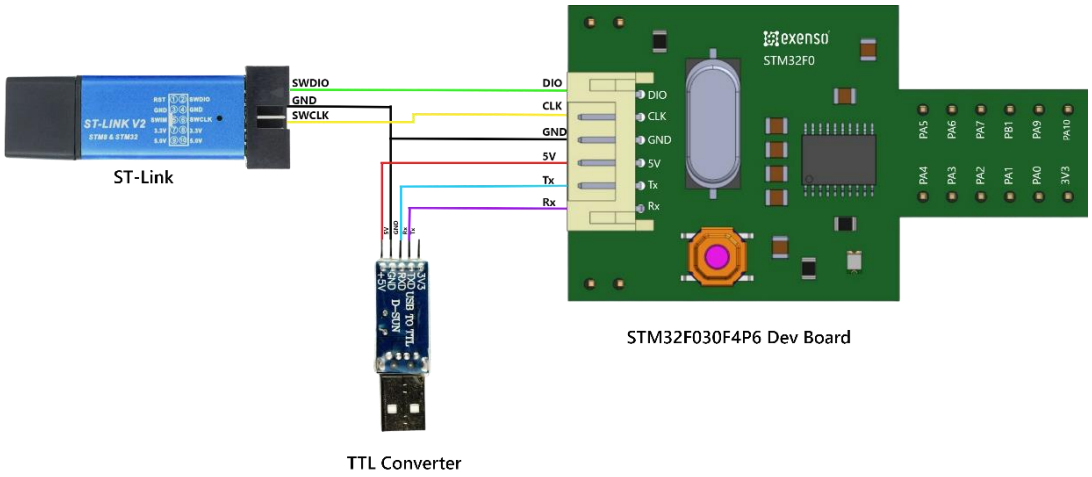
## Components Required

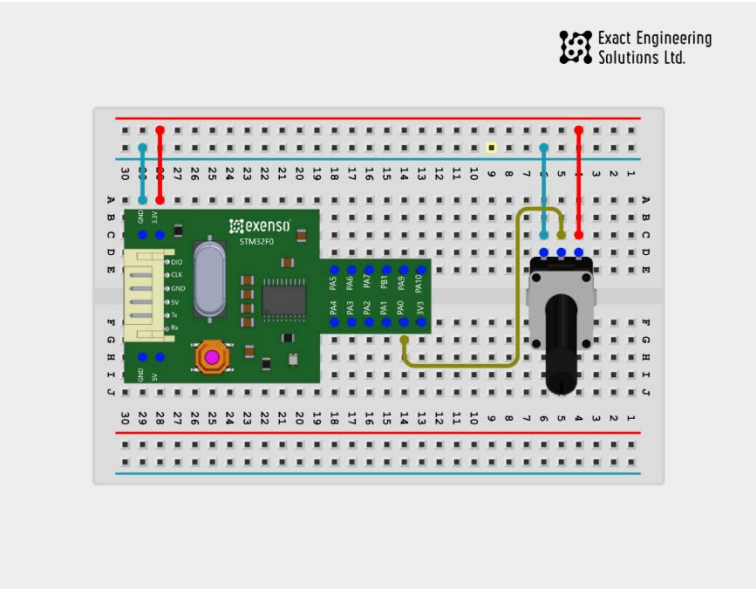
You will need the following components –

- 1 × Breadboard
- 1 × STM32F030F4P6
- 1× TTL Converter
- 1× 10KΩ potentiometer
- Some Jumper wire

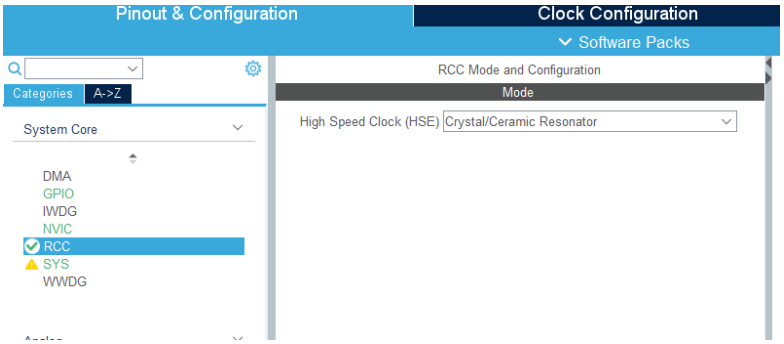
## Procedure

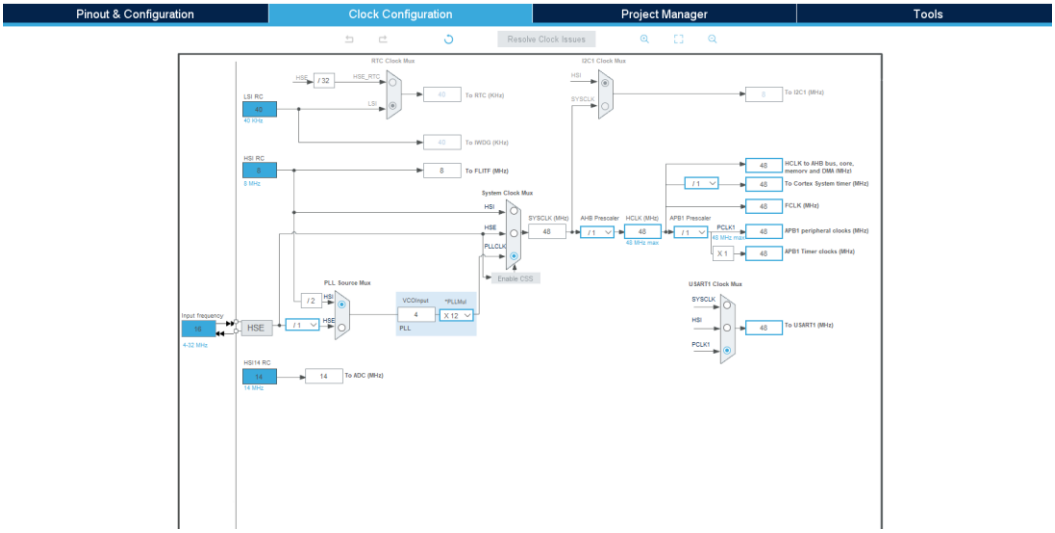
Follow the circuit diagram shown in the image given below.

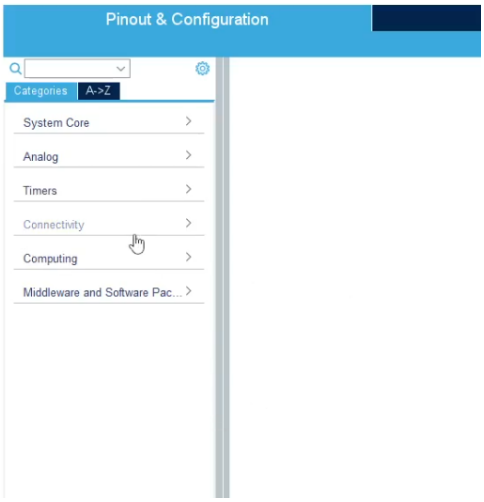


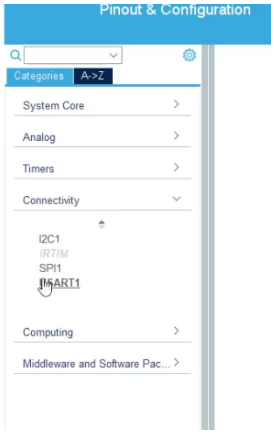


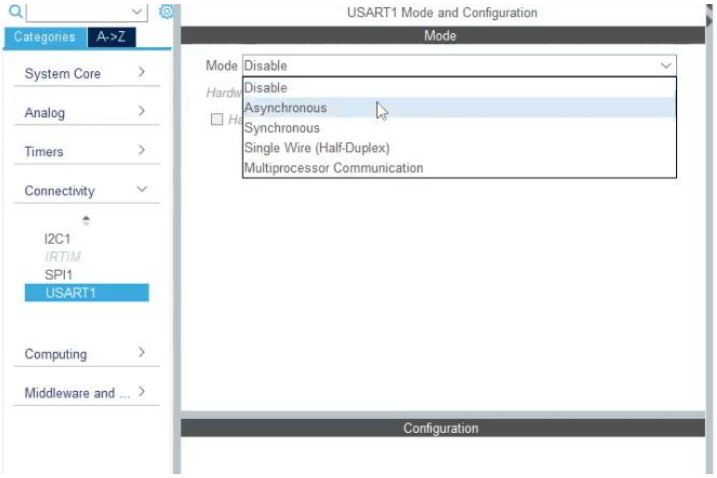

# STM32F0 Pin Configuration:



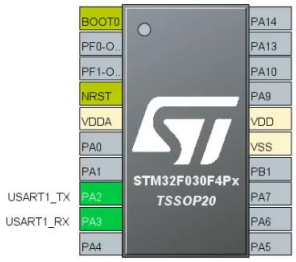



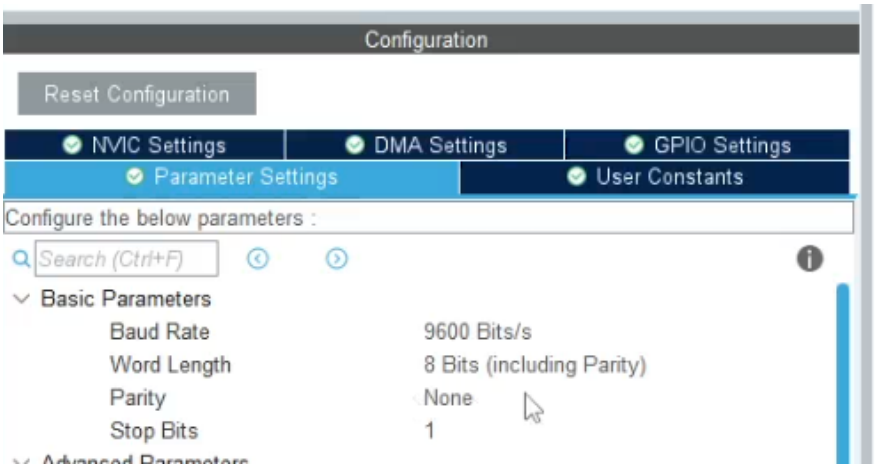

Pinout & Configuration
Categories
A->Z
System Core
Analog
Timers
Connectivity
Computing
Middleware and Software Pac...

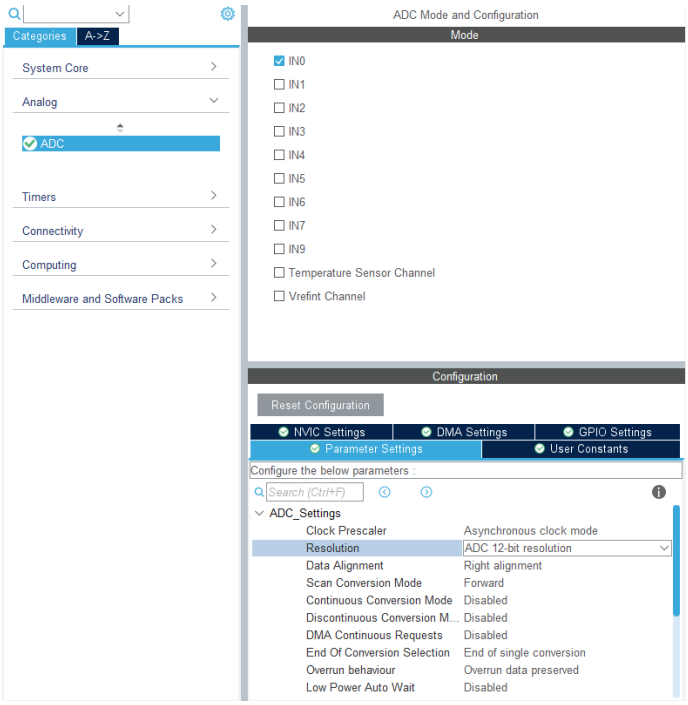

Pinout & Configuration
Categories
A->Z
System Core
Analog
Timers
Connectivity
I2C1
IRTIM
SPI1
USART1
Computing
Middleware and Software Pac...

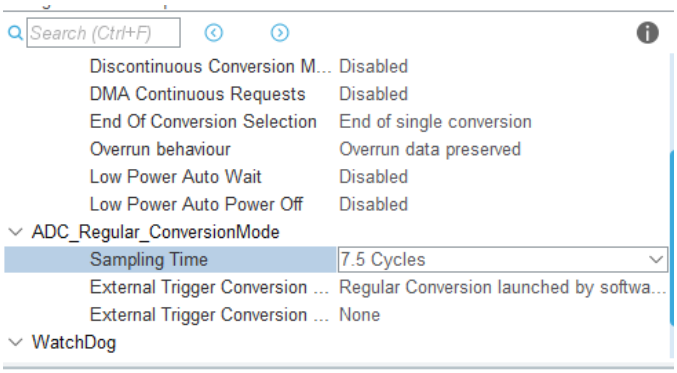

Categories
A->Z
System Core
Analog
Timers
Connectivity
I2C1
IRTIM
SPI1
USART1
Computing
Middleware and ...
USART1 Mode and Configuration
Mode
Mode
Disable
Disable
Asynchronous
Synchronous
Single Wire (Half-Duplex)
Multiprocessor Communication
Configuration

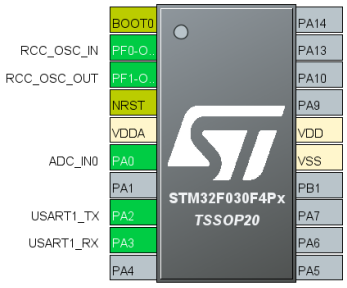

Mode Asynchronous
Hardware Flow Control (RS232) Disable
Hardware Flow Control (RS485)
Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
Basic Parameters
Baud Rate 38400
Word Length 8 Bits (including Parity)
Parity None
Stop Bits 1
Advanced Parameters
Data Direction Receive and Transmit


BOOT0
PF0-O..
PF1-O..
NRST
VDDA
PA0
PA1
USART1_TX PA2
USART1_RX PA3
PA4
PA14
PA13
PA10
PA9
VDD
VSS
PB1
PA7
PA6
PA5
STM32F030F4Px
TSSOP20


Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
Basic Parameters
Baud Rate 9600 Bits/s
Word Length 8 Bits (including Parity)
Parity None
Stop Bits 1


Categories
A->Z
System Core
Analog
ADC
Timers
Connectivity
Computing
Middleware and Software Packs
ADC Mode and Configuration
Mode
IN0
IN1
IN2
IN3
IN4
IN5
IN6
IN7
IN9
Temperature Sensor Channel
Vrefint Channel
Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
ADC_Settings
Clock Prescaler
Asynchronous clock mode
Resolution
ADC 12-bit resolution
Data Alignment
Right alignment
Scan Conversion Mode
Forward
Continuous Conversion Mode
Disabled
Discontinuous Conversion M...
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
End of single conversion
Overrun behaviour
Overrun data preserved
Low Power Auto Wait
Disabled

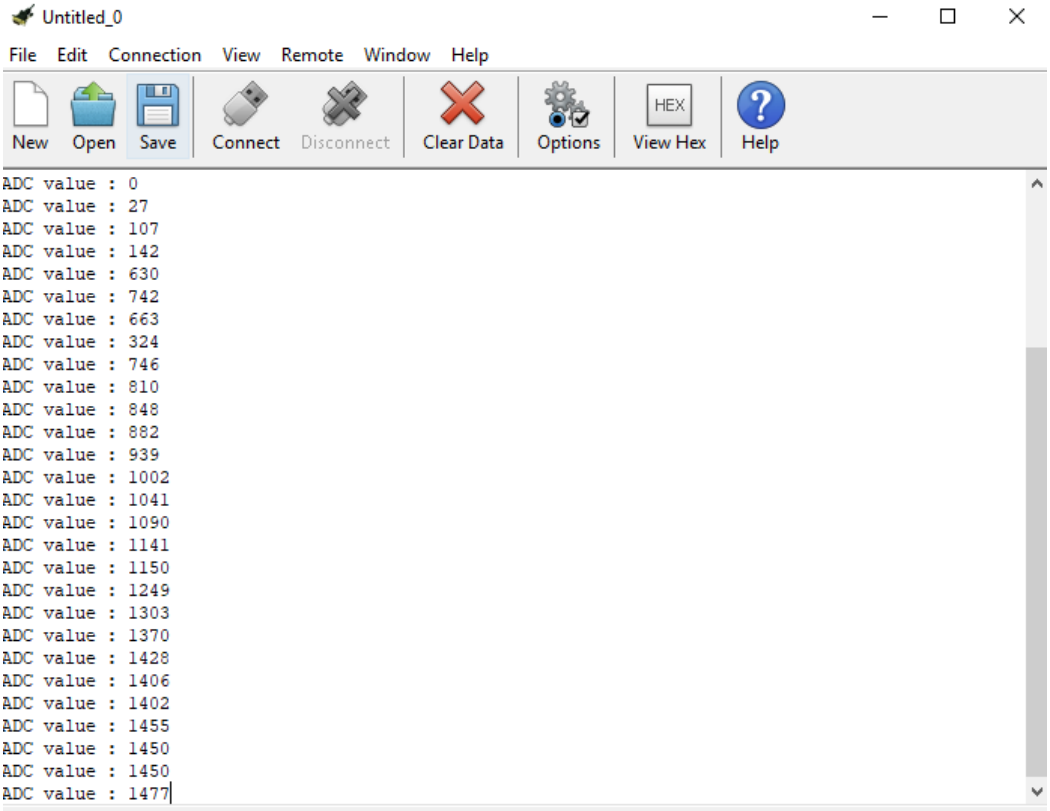

Search (Ctrl+F)
Discontinuous Conversion M...
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
End of single conversion
Overrun behaviour
Overrun data preserved
Low Power Auto Wait
Disabled
Low Power Auto Power Off
Disabled
ADC_Regular_ConversionMode
Sampling Time
7.5 Cycles
External Trigger Conversion ...
Regular Conversion launched by softwa...
External Trigger Conversion ...
None
WatchDog


BOOT0
RCC_OSC_IN
PF0-O..
RCC_OSC_OUT
PF1-O..
NRST
VDDA
ADC_IN0
PA0
PA1
USART1_TX
PA2
USART1_RX
PA3
PA4
PA14
PA13
PA10
PA9
VDD
VSS
PB1
PA7
PA6
PA5
STM32F030F4Px
TSSOP20

## Code

```c
#include "main.h"

#include "Print_Number.h"

ADC_HandleTypeDef hadc;

UART_HandleTypeDef huart1;
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_ADC_Init(void);
static void MX_USART1_UART_Init(void);


uint16_t ADCvalue = 0;     // Init int value
uint8_t adc_buff[4];
uint8_t next_line[20] ="\nADC value : "; // next line



int main(void)
{
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_ADC_Init();
  MX_USART1_UART_Init();

  while (1)
  {

      HAL_ADC_Start(&hadc);
      HAL_ADC_PollForConversion(&hadc,200);
      ADCvalue = HAL_ADC_GetValue(&hadc);

       // convert ADC int to string
      int_to_str(ADCvalue,adc_buff);
      // for next line
      HAL_UART_Transmit(&huart1, next_line,strlen(next_line), 1000);
      // Print ADC Value
      HAL_UART_Transmit(&huart1, adc_buff, strlen(adc_buff), 1000);
      // Delay 1000ms or 1s
      HAL_Delay(1000);
  }

}
```

## Output:


Untitled_0
File Edit Connection View Remote Window Help
New Open Save Connect Disconnect Clear Data Options View Hex Help
ADC value : 0
ADC value : 27
ADC value : 107
ADC value : 142
ADC value : 630
ADC value : 742
ADC value : 663
ADC value : 324
ADC value : 746
ADC value : 810
ADC value : 848
ADC value : 882
ADC value : 939
ADC value : 1002
ADC value : 1041
ADC value : 1090
ADC value : 1141
ADC value : 1150
ADC value : 1249
ADC value : 1303
ADC value : 1370
ADC value : 1428
ADC value : 1406
ADC value : 1402
ADC value : 1455
ADC value : 1450
ADC value : 1450
ADC value : 1477